J Sign Process Syst (2010) 60:131–145
DOI 10.1007/s11265-009-0408-6

Methods for Power/Throughput/Area Optimization of H.264/AVC Decoding

Ke Xu · Tsu-Ming Liu · Jiun-In Guo · Chiu-Sing Choy

Received: 23 August 2008 / Revised: 11 August 2009 / Accepted: 8 September 2009 / Published online: 20 October 2009

Abstract This paper presents methods for efficient optimization of ASIC implementation for H.264/AVC video decoding. A systematic approach in optimization is presented in a top-down flow. Tradeoffs among Power, Throughput, and Area (PTA) at both system level and block level are studied and balanced. The system architecture is first evaluated. We then focus on the pipeline organization, parallelism, and memory architecture optimization. Different pipeline granularities are compared and their pros-and-cons are evaluated. Various parallel scenarios, especially 1×4-column and 4×1-row, are analyzed and compared. Then the detailed designs of various building blocks, such as inverse transform, inter prediction, and deblocking filter, are evaluated and their intrinsic characteristics are exploited to facilitate PTA optimization. Finally, we provide the design guidelines for ASIC implementation based on the analysis and our design experiences of five dedicated decoder chips.

Keywords ASIC · Cost · Decoding · H.264/AVC · Memory · Performance · Power

K. Xu (✉) · C.-S. Choy
The Chinese University of Hong Kong,
Hong Kong, P.R.China
e-mail: kexu@ee.cuhk.edu.hk

C.-S. Choy
e-mail: cschoy@ee.cuhk.edu.hk

T.-M. Liu
National Chiao Tung University,
Hsinchu, Taiwan
e-mail: mingle@royals.ee.nctu.edu.tw

J.-I. Guo
National Chung Cheng University,
Min-Hsiung, Taiwan
e-mail: jiguo@cs.ccu.edu.tw

1 Introduction

Due to the rapid advances in VLSI technology, algorithms that have been regarded to be too complex to realize becomes manageable. This in turn encourages developers to continue developing more and more complex algorithms. One such example is video codec, which is extremely complicated with very high processing and memory access rates. Evolving from early H.261/MPEG-1, the latest H.264/AVC [1, 2] coding standard is able to provide nearly two times coding efficiency compared with its predecessors such as MPEG-2 or H.263, at the cost of much higher complexity.

Compared with encoder, the decoder has wider applications ranging from high-resolution set-top boxes of HDTV to low-resolution display on mobile or hand-held devices. Ideally, all decoder designs should achieve the best in all three performance parameters, power, throughput, and area (PTA). In reality, different applications will demand specific requirements on the final implementation. Focusing primarily on performance for HDTV will result in too much power consumption. Focusing only on energy for mobile applications is equally inadequate since it may not achieve the required performance. The correct approach is a joint optimization of PTA that balances the tradeoffs among them subject to specific application constraints.

In general, there are many methods to realize such a video codec system. Basically, what an optimal architecture should be depends on applications and the overall system considerations. The many methods can be roughly divided into three categories: general-purpose processors [3], DSP processors [4], and hardwired/dedicated decoders [5–14]. Using a general purpose processor, it is difficult to achieve real-time performance even operating at a very high frequency since the general

data path and sequential processing flow are not compatible with video processing. Special multimedia/video processors or DSP may be a better solution, but either an instruction set with parallel processing extensions, or dedicated hardware accelerators are normally required. Furthermore, they are not power/area efficient since they are designed for a wide range of applications. In contrast, a fully dedicated video codec architecture is the best choice to obtain optimized performance since one can carefully exploit the operation characteristics to fine tune each module. ASIC designs usually make use of a highly parallel and/or pipelined architecture to lower the operating frequency [15].

Although designs of dedicated decoders have been discussed in [5–14], they are designed for specific applications. There is no work in literatures ever systematically studying the joint optimization of PTA and how to balance design tradeoff for different applications. Based on the design experience of several full-function chips [5–9], this paper proposes design methods and guidelines at both system and module levels for efficient video decoding under different application constraints.

The rest of this paper is organized as follows. Decoder architecture, including basic H.264/AVC algorithm, system and pipeline architectures is described in Section 2. An efficient memory organization is presented as well. Joint PTA optimizations for various building blocks are discussed in Section 3. Video decoder design guidelines are proposed in Section 4. Finally, we conclude the paper in Section 5.

2 Decoder Architecture

2.1 Algorithm

In general, video coding algorithms contain hybrid coding tool sets which work in concert to reduce the temporal, spatial, psycho-visual, and statistical redundancies in raw video signals. General hybrid-video coding structure in H.264/AVC is graphically depicted in Fig. 1. A more detailed description can be found in [1].

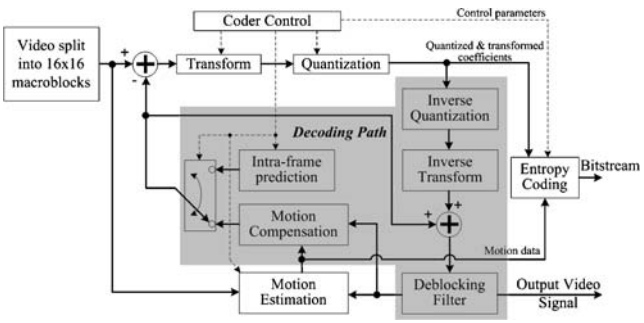


Figure 1 Hybrid video coding diagram.

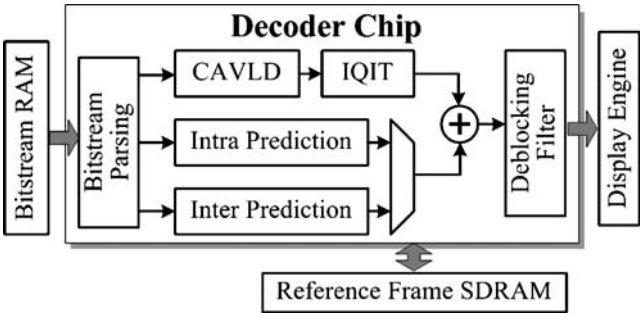


Figure 2 Decoder system realization.

Unlike the encoder where designers can choose different algorithms favoring one or several aspects of PTA, the decoding algorithm is fully specified in the standard and there is not much flexibility designers can play around. Design [16] is the only example where decoder PTA is optimized at algorithm level. It modifies the original interpolation operation from a 6-tap FIR filter to a 4-tap diagonal filter to reduce both the operating power and circuit area. However, the filtered output deviates from what is expected from the standard and thus the decoder is not a fully standard-compliant solution.

2.2 System Architecture

A decoder architecture can be obtained by mapping the blocks in the decoding paths in Fig. 1 into functional units and hard-wiring the communications among them. The direct mapping has several advantages. First, each block is thus small and easy to design and implement. Second, the interfaces among the different blocks are simple and easy to manage. Third, the blocks can be optimized individually according to their characteristics. For example, the syntax parser is characterized with a lot of control-sensitive paths whereas the inter prediction with huge computation complexity and memory access. Therefore, optimization of power and area is of the first priority for syntax parser, while optimization of power and throughput is more a necessity for inter prediction. As in most of the designs [5–9], a decoder system is assumed to be mapped into several building blocks such as bitstream parsing, IQIT (Inverse Quantization Inverse Transform), intra prediction, inter prediction, and deblocking filter, as illustrated in Fig. 2.

2.3 Pipelining

Since the complexity of video decoding is continuously increasing, serial processing definitely can not satisfy the real time decoding requirements. Therefore, pipelining, which is a well-known technique for exploiting temporal concurrency, is indispensable for H.264/AVC decoding. In

general, there are three kinds of pipelining granularity, 16×16, 8×8, and 4×4. A fine granularity pipeline (4×4/8×8 block level) requires fewer registers but introduces many stalls or bubbles frequently at 4×4/8×8 level, leading to added processing cycles for one MB. On the other hand, a coarse granularity pipeline (usually 16×16 MB level) can reduce the pipeline stalls at the cost of additional pipeline registers. Figure 3 graphically compares a 4×4 block pipeline with a 16×16 MB pipeline. For both granularity, the decoder task is divided into three parts, residual decoding (inverse transform and inverse quantization), prediction (inter and intra prediction), and deblocking filter.

The cycle time for a single stage in the 4×4 block pipeline equals the longest cycle (S_{max}) among current S1, S2, and S3 since the pipeline needs to be synchronized every 4×4 block. Assuming a 4:2:0 chrominance format, the total time for decoding a macroblock equals 24 (16 luma, 4 Cb, and 4 Cr) times S_{max} as:

$$MB\ average_{4\times4pipeline} = 24 \times \max(S1(i), S2(j), S3(k)) \tag{1}$$

As for the MB pipeline, 24 4×4 blocks during S1 ~ S3 stages are handled altogether before three MBs in different macroblock pipeline stages are synchronized. All the 4×4 blocks inside the same MB are processed continuously without any stall. The pipeline bubbles are "pushed" to the end of pipeline and take up less time than the sum of bubbles in the 4×4 block pipeline since they may cancel out each other. Therefore, the MB pipeline is more efficient in terms of system throughput.

$$MB\ average_{16\times16pipeline} = \max\left(\sum_{i=0}^{23} S1(i), \sum_{j=0}^{23} S2(j), \sum_{k=0}^{23} S3(k)\right) \tag{2}$$

The 8×8 pipeline will be a compromise between the 4×4 and 16×16 pipelines. A more convincing processing cycle comparison of various granularities is described in Table 1 from [18]. Compared with the 4×4 block pipeline, the 16×16 MB pipeline improves the throughput by 26% at the cost of additional hardware.

The estimated operating frequency for different pipeline granularities under various video resolutions (all 30fps) is described in Table 2. Based on cycle-accurate simulation, 4×4 pipeline takes 500 cycles to process one macroblock in average. Therefore, the required frequency for 1080p decoding is around 122 MHz (500×8160 MB/frame × 30frame/s). Take the factor "1.26" from Table 1, the required frequency for the same 1080p decoding under 16×16 granularity is around 97 MHz (122 MHz/1.26).

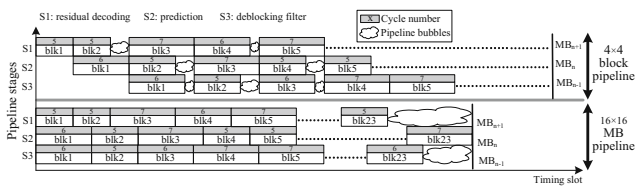


Figure 3 4×4 block pipeline vs. 16×16 MB pipeline.

Table 1 Processing cycles comparison.

	16×16	8×8	4×4
Cycles/MB	1unit	1.19unit	1.26unit

To verify the accuracy of these estimations, we compare several estimated operating frequencies with real chip measured frequencies in Table 3, proving our estimation is accurate within ±5% range. It is worth noting that most of the recent works do not employ single granularity pipeline architecture, and the operating frequency is limited by the most-critical building block and the pipeline granularity adopted.

Generally, there are always design tradeoffs among PTA for various video applications. To look at their individual impact closely, throughput, area, and power curves of different pipeline granularities with respect to video resolution are plotted in the following.

The throughput curve is rather simple. Because no matter what the pipeline granularity is, it needs to guarantee a minimum but fixed output rate for real-time display. When the video resolution goes up, the required overall throughput (Mpix/s) rises accordingly, but the normalized Mpix/MHz throughput which measures the characteristics of a pipeline does not scale accordingly, as plotted in Fig. 4.

The heuristic logic area (not including on-chip SRAMs) tradeoff curve is plotted in Fig. 5. The area is derived based on logic synthesis results from Design Compiler targeting 180 nm technology. The synthesis constrain frequency is the same as Table 2. It can be seen how the increase in decoding frequency affects the area of the decoder. For low resolution decoding, fine granularity is more area-efficient due to the reduced number of intermediate registers. When the video resolution scales up, fine granularity pipeline decoder needs more effort to size up cells to satisfy the realtime decoding requirement. The coarser granularity becomes advantageous in terms of area after the crossing point of the corresponding curves.

The heuristic power tradeoff curve is plotted as well, and only dynamic power, which is proportional to cv^2f, is taken into consideration. When the video resolution increases, the power consumption rises due to the increase of both decoder area and operating frequency. At lower video resolutions, all the three granularities have nearly the same power dissipation. Although the fine granularity has smaller

Table 2 Estimated operating frequency.

	QCIF 176×144 (90kb/s)	D1 720×576 (1.3Mb/s)	HDTV720p 1280×720 (2.7Mb/s)	HDTV1080p 1920×1088 (8Mb/s)
4×4	1.49 MHz	24.3 MHz	54 MHz	122 MHz
8×8	1.4 MHz	22.9 MHz	51 MHz	116 MHz
16×16	1.18 MHz	19.3 MHz	42.9 MHz	97 MHz

design area (smaller capacitance, c), it demands a relatively higher speed (larger frequency, f). At higher video resolution, the coarse granularity benefits from both smaller area and lower operating frequency, which leads to smaller power consumption, thus the separation of three curves (Fig. 6).

Based on the above analysis, we can intuitively conclude that for low resolution applications, fine granularity pipeline is more efficient in terms of both area and power, while for high resolution, coarse granularity is more preferable.

Many conventional designs utilize the uniform pipeline architecture without considering task variations among different building blocks, which may lead to extra pipeline registers and power. A hybrid pipeline architecture that adopts different pipeline granularities for various coding tools is first proposed in [19] and gradually becomes the dominant pipelining architecture for H.264/AVC video decoding. Usually, for low video resolutions (QCIF, CIF, etc.) or the non-critical paths in high video resolution (Intra prediction, CAVLD, etc.), 4×4/8×8 fine granularity pipeline is preferable to save hardware cost and power, while for the critical paths (Inter prediction, deblocking filter, etc.), 8×8/16×16 coarse granularity pipeline is preferable to satisfy real-time requirement.

2.4 Parallelism

In addition to pipelining, parallelism is widely adopted to improve the throughput and reduce operating frequency. Serial processing with 1pixel/cycle throughput can not satisfy the realtime decoding requirement. Simulation results show that around 1300 cycles are required to decode one 16×16 macroblock if pixels are processed in serial, which leads to an unusually high frequency of 310 MHz for a HD1080 dedicated decoder. On the other hand, predicting all the 16 pixels in a single clock cycle leads to larger chip area and more standby power. This also tends to be an over-design since the syntax parsing, which is control-sensitive and needs to be synchronized with prediction, cannot run as fast. In addition, the predicted results from previous cycles, as well as neighboring datum cannot be reused. The intermediate registers that store predicted results will also increase. Based on the above analysis, most designs adopt a throughput of predicting 4 pixels in one clock cycle, which is a compromise between hardware cost and system performance. The processing sequence of 4 pixels can be either 1×4 in a column or 4×1 in a row, as compared in Table 4.

Although the calculation complexity is the same for all modules under both 1×4 column and 4×1 organization, memory access efficiency and hardware cost are different. Due to double-z zig-zag order of decoding, 1×4 column helps to share more reference pixels during horizontally 4×4 block switching, thus the number of memory access can be reduced as compared with 4×1 row. For example, during intra prediction, the last (right-most) 1×4 column of a 4×4 block can be directly used as the reference for its right neighboring 4×4 block. However, if we utilize 4×1 row method, such kind of reference pixel reuse is totally lost since the predictions of two horizontal adjacent 4×4 blocks are never abutting. As for hardware cost in sum module, the output of 4×1 row can be directly delivered to reference frame which is also organized horizontally, while the output of 1×4 column has to be undergo column-to-row switching before written out. Therefore, a hybrid parallel scenario for different modules is recommended to achieve best performance. More detailed analysis can be found in [17].

2.5 Memory Organization

As aforementioned, memory access is one of the bottlenecks in H.264/AVC decoding since predicting the current pixel needs a significant amount of neighboring data from

Table 3 Frequency estimation accuracy.

	Granularity	Resolution	Chip measured	Estimated from Table2	Variation
[5]	4×4	QCIF	1.5 MHz	1.49 MHz	−0.67%
[7]	4×4	HDTV 1080p	120 MHz	122 MHz	+1.67%
[6]	16×16	HDTV 1080p	100 MHz	97 MHz	−3%

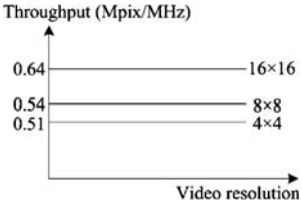


Figure 4 Normalized throughput curve.

memory, and the predicted results need to be stored back as well. Memory power may account up to 70% of the whole decoder power dissipation. If all the data communication occurs directly between the chip core and the off-chip SDRAM, the memory power consumption may be formidable. Fortunately, the H.264/AVC standard is identified with a peculiar locality access; there is a high probability that spatially adjacent pixels in both horizontal and vertical directions are accessed. This locality can be exploited to design a cache-like memory hierarchy for video decoding. The principal idea is to buffer highly correlated data from off-chip or large memories to smaller on-chip ones, which helps to shift a great portion of memory access to less power-demanding memories while maintaining the same performance.

The memory requirement of neighboring data is tabulated in Table 5. We assume 4:2:0 chroma sampling and W is the picture width in pixels. To have a concrete view of each process's requirements, memory sizes for two representative resolutions, QCIF (176x144) and HDTV1080 (1920x1088) are also described.

We found that the most critical memory is for deblocking filter raw pixels (64 W) and intra neighboring pixels (16 W). Other kinds of memory can be easily handled by on-chip SRAM or register files.

Generally, there are three possible schemes to implement the (64 W+16 W) memory. They are illustrated in the following sub-sections.

2.5.1 NoC (Non-on-Chip)

The NoC scheme assigns all the reference data to off-chip SDRAM. Only a very small number of information is stored on-chip [20–23]. There is frequent retrieval or

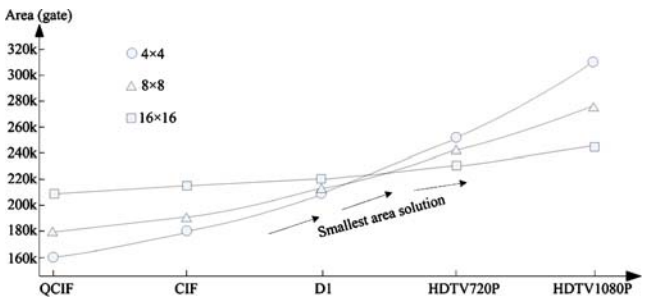


Figure 5 Logic area curve.

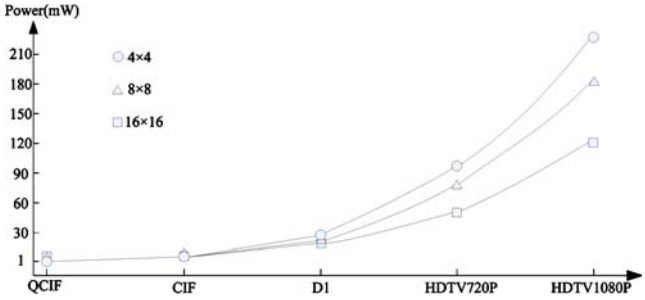


Figure 6 Power curve.

updating of internal data from external memory. In general, the on-chip memory size can be reduced significantly at the cost of increased external memory bandwidth. It is worth noting that the reduction of on-chip memory access is larger than the increase in off-chip memory bandwidth, since not all the reference data stored off-chip needs to be read back as reference again [7].

2.5.2 FoC (Full-on-Chip)

The FoC scheme stores all the reference data on-chip [12, 24]. The size of the memory is proportional to the frame width. This is negative property when the video resolution is increased. It costs a significant amount of chip area, but the external SDRAM access bandwidth and I/O power are reduced to the minimum. There do exist some redundant memory write operations in this scheme. For example, since there is no way of knowing whether a macroblock of the next row is to be intra predicted, the intra upper neighboring pixel memory should always been updated.

2.5.3 PoC (Partial-on-Chip)

The PoC scheme is a compromise between NoC and FoC. The key idea is that not all of the neighboring data should be stored on-chip. In a certain sequence or bit rate, the prediction or filtering of most of the edges can be skipped. Therefore, there is no need to keep them for the following decoding process and thus the size of on-chip memory can be reduced. The critical design decision is to determine the size of the on-chip memory and decide which part of reference data should be stored. Design [25] proposed a Line-Pixel-Lookahead scenario to predict the size of the buffer.

The PTA requirements for the above three schemes are compared in the following. Some terminologies are defined below for ease of understanding:

W: video picture width in pixel
H: video picture height in pixel
F: frame rate

Table 4 Comparison of 1×4 column method and 4×1 row method.

		IQIT	Intra pred.	Inter pred.	Sum	Deblock-ing filter
Hardware cost	4×1	same	same	same	no buffer	no effect
	1×4				extra buffer 16 × 8bit	no effect
Memory access efficiency	4×1	same	100%	100%	same	no effect
	1×4		83.3%	72%		no effect
Throughput	4×1	same	lower	lower	higher	no effect
	1×4		higher	higher	lower	no effect
Calculation complexity	4×1	same	same	same	same	no effect
	1×4					no effect
Overall	4×1	same	×	×	√	no effect
	1×4	same	√	√	×	no effect

In addition, we make several assumptions to facilitate comparison:

1) 4:2:0 format for chroma sub-sampling.
2) For PoC scheme, we assume 1/8 memory size compared with FoC, and 60% missing rate according to [25].
3) The percentage of Intra prediction is 1/10 inside the video sequence.
4) 60% edges with Bs = 0 for deblocking filter [26].

The FoC approach requires 80 W bits, and since the PoC approach reduces the SRAM size by 1/8, it only needs 10 W bits. NoC stores nearly all the reference information to off-chip SDRAM, thus there is no need for on-chip SRAM (Table 6).

The memory bandwidth is estimated based on the aforementioned assumptions. Take the FoC scheme as an example, the memory bandwidth is calculated as follows:

1) SRAM write for Intra prediction:

$$\frac{W \times H}{256} \times 16 \times 2 \times 8bit \times F = WHF \tag{3}$$

2) SRAM write for deblocking filter:

$$\frac{W \times H}{256} \times 16 \times 2 \times 32bit \times F = 4WHF \tag{4}$$

So the total SRAM write bandwidth is 5WHF bit/s.

3) SRAM read for Intra prediction:

$$\frac{W \times H}{256} \times 16 \times 2 \times 8bit \times 10\% \times 75\% \times F = 0.075WHF \tag{5}$$

4) SRAM read for deblocking filter:

$$\frac{W \times H}{256} \times 16 \times 2 \times 32bit \times 40\% \times F = 1.6WHF \tag{6}$$

So the total SRAM read bandwidth is 1.675WHF bit/s.

As can be seen, SRAM read is only 33.5% of SRAM write, which means totally 66.5% SRAM write is redundant. This is because during SRAM write period, the decoder does not know the decoding information of the next macroblock row, thus there are substantial redundant write operations. The overall memory bandwidth comparison is illustrated in Table 7.

Table 5 Neighboring data requirement.

Block	Left	Upper		
		Formula	QCIF	HDTV 1080
CAVLC nC	40b	5 W/2	440b	4.8Kb
Intra prediction mode	16b	W	176b	1.9Kb
Intra neighboring pixel	256b	16 W	2.8Kb	30.7Kb
Inter motion vector	64b	4 W	704b	7.68Kb
Deblocking filter	1Kb	64 W	11.2Kb	122.9Kb
Total	1.4Kb	87.5 W	15.4Kb	168Kb

Table 6 Memory size comparison.

	NoC	PoC	FoC
On-chip SRAM	0	10 W bits	80 W bits
Off-chip SDRAM	12WH bit	12WH bits	12WH bits

To facilitate comparison, we assume the SRAM power consumption is 1unit. Because the I/O and SDRAM power is usually larger than SRAM [27], we can assume the I/O power is 1 × a unit and the SDRAM power is 1 × a × b unit (a>1, b>1). The memory power comparison is summarized in Table 8.

Figure 7 plots the contours of three schemes total system power. Curves belonging to the same contour line group are of the same power number. For small a and b, three schemes do not exhibit huge difference. As a and b get larger, contour lines become more and more separated which means the power of three schemes varies significantly. In order to have a quantitative analysis of which scheme performs best in terms of power consumption under different conditions, Table 9 and Table 10 are derived from Table 8.

Regarding the chip power, if the I/O power consumption is considerably large (a>5) compared with SRAM, FoC is the best solution since it reduces I/O access to the minimum. Otherwise, PoC is the most power-efficient since it tries to balance off the SRAM power and I/O power.

Regarding the system power, the point at which FoC being the lowest power solution is even lower. Due to the relatively large SDRAM power (large a and b), a(1 + b) can easily exceed 5.375unit.

3 Building Blocks

PTA optimization at system level concerns with pipelining, parallelism, and memory organization. In order to achieve a globally optimized design, it is essential to push the PTA optimization across the design boundary, from higher system level to lower block level. This section presents the PTA optimization and circuit implementation of several main building blocks in a H.264/AVC decoder.

3.1 Inverse Transform

H.264/AVC adopts transform coding for the prediction residues. The transformation is applied to a 4×4 block, and is an integer orthogonal approximation of the traditional Discrete Cosine Transform (DCT). Each 4×4 block can be directly computed with direct 2D approach, or using separate 1D approach with a transposition buffer. To further reduce the computation complexity, the fast butterfly algorithm was proposed in [28]. An implementation example of butterfly with direct 2D is depicted in Fig. 8.

The most straightforward implementation for inverse transform is a set of adders in serial, as shown in Fig. 9.

To reduce the hardware cost, we can "fold" the BU of Fig. 9 and use DFF to hold the intermediate result, as depicted in Fig. 10. The 3-folding architecture takes 3 clock cycles for single pixel calculation, but reduces the number if BU from 3 to 1.

If performance is of the first priority like in HDTV applications, we can process all the 16 pixels of a 4×4 block in parallel, as depicted in Fig. 11. Such scheme is able to deliver 1 block/cycle throughput. However, the hardware cost is too high and the data path is too long (15 adders in a single clock cycle). Fast butterfly and folding can be utilized to reduce the complexity and hardware cost.

All the approaches mentioned above, folding, butterfly, etc., can be combined to satisfy a certain PTA constraint. They are evaluated in Table 11.

Some comments of the comparison are:

1) We evaluate relative energy instead of power since the throughput of each implementation varies significantly.
2) The energy is estimated based on the number of additions and the number of register read/write operations. According to [29], add operations consume approximately 3× energy more than register operations with the same bit size. Therefore, the relative energy

Table 7 Memory bandwidth comparison (bit/s).

		NoC	PoC	FoC
On-chip SRAM	Read	0	1.675WHF × 40%	1.675WHF
	Write	0	1.675WHF × 40%	5WHF
	Total	**0**	**1.34WHF**	**6.675WHF**
Off-chip SDRAM	read	1.675WHF	1.675WHF × 60%	0
	write	12WHF	12WHF	12WHF
	Total	**13.675WHF**	**13.005WHF**	**12WHF**

Table 8 Memory power comparison.

	NoC	PoC	FoC
On-chip SRAM	0	1.34	6.675
I/O	13.675a	13.005a	12a
Total chip power	**13.675a**	**1.34 + 13.005a**	**6.675 + 12a**
Off-chip SDRAM	13.675ab	13.005ab	12ab
Total system power	**13.675a(1 + b)**	**1.34 + 13.005a(1 + b)**	**6.675 + 12a(1 + b)**

for 1 register operation is 1 unit while for 1 add operation is 3 units.

3) The critical path delay is evaluated in terms of additions in a single clock cycle.
4) The throughput is inversely proportional to the latency:

$$Throughput \propto \frac{1}{latency} = \frac{1}{(\#of\ cycles) \times (critical\ path\ delay)} \quad (7)$$

Figure 12 summarizes energy-throughput-area characteristics of different approaches. Figure 12 1) shows that since the parallel approach has a higher throughput, the supply voltage can scale down to reduce the power consumption. The same applies to the direct 2D approach. Figure 12 2) shows that no matter folding is used or not, the throughput (product of # of cycles and critical path delay) is the same. Folding method trades a small area for high energy as a result of additional register access. The only approach that optimizes all the three performance parameters is the butterfly algorithm, as illustrated in Fig. 12 4).

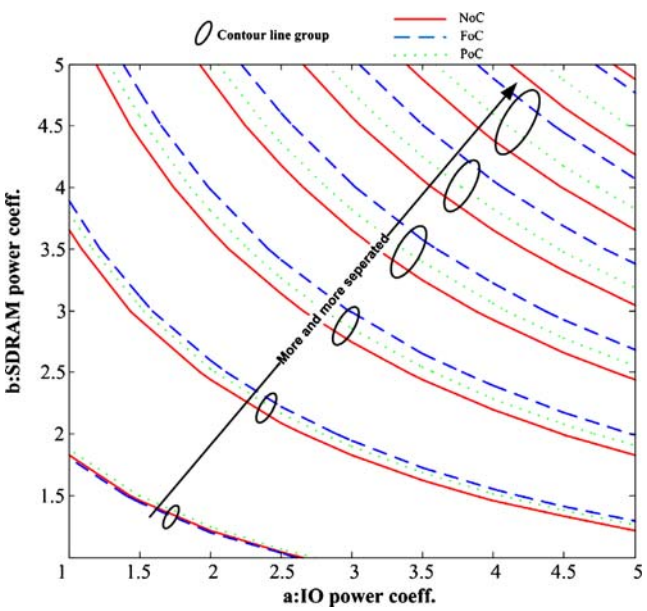


Figure 7 Three schemes contour plots.

3.2 Inter Prediction (Motion Compensation)

The inter prediction, as well as the deblocking filter, is the bottleneck of the entire decoder system in terms of memory access and computation complexity. For current prediction, huge amount of reference pixels should be fetched from external memory and stored on chip. According to Motion Vector (MV) position, reference pixels are interpolated by different tap of filters to obtain predicted value. In this subsection, we will mainly focus on memory bandwidth optimization, throughput improvement, as well as hardware resource sharing, to improve the PTA of the inter prediction.

3.2.1 Adaptive Pipeline

During inter prediction, different MB partitions and motion vectors require different number of memory accesses and pixel computations. Therefore, the pipeline should be adaptive to reduce unnecessary stalls and improve throughput. The adaptive pipeline can be roughly divided into two macro-stages, reference pixel fetch and interpolation [30].

3.2.2 Memory Bandwidth Optimization

Simulation results show that the memory bandwidth is up to 528 MB/s and 878 MB/s for decoding H.264/AVC baseline and main profile video, respectively. Since most of the bandwidth is consumed by inter prediction, bandwidth optimization is of first priority. We propose several techniques in the following.

Variable block fetch The realization of inter prediction with quarter-sample motion vectors in reference software JM [31] fetches reference data in unit of 9×9 for each 4×4

Table 9 Chip power evaluation (excluding SDRAM).

	a < 1.985unit	2 unit < a < 5.33 unit	5.33unit < a
Min. power	NoC	PoC	FoC

Table 10 System power evaluation (including SDRAM).

	a (1 + b) < 2unit	2 unit < a(1 + b) < 5.375 unit	5.375unit < a(1 + b)
Min. power	NoC	PoC	FoC

block, without eliminating any redundant data access. However, experimental results show that there is a large probability that not all the 9×9 block is necessary for prediction. Design [7] proposed a Variable Block Size (VBS) scheme to provide the flexibility to respectively fetch reference data in units of 13×13, 13×9, 9×13, and 9×9 pixels. Design [30] proposed a Variable Block Shape (VBSH) approach to further reduce redundant memory access according to individual pixel positions, as illustrated in Fig. 13.

Compared with straightforward memory access which always involves loading of 9×9 reference blocks, the VBSH reduces reference pixel access by 33.6% in average.

Buffer reuse According to the experimental results, the inter prediction operations using larger blocks sizes occur more frequently than those using smaller block sizes. Therefore, to increase data reuse and reduce memory bandwidth, a larger buffer is preferable in terms of data reuse.

Figure 14 shows an example of 4×8 block mode to exploit data reuse of neighboring 4×4 block prediction. Other modes like 8×4, 8×8, or even larger, can be derived similarly. It is proved that the larger the partition, the more reference pixels can be shared among neighboring blocks in comparison with 4×4 partition. In summary, the VBSH scheme eliminates redundant reference pixel fetch while buffer reuse helps to reduce the overlapped memory accesses. These two techniques totally contribute to 33.6%, 59.3%, and 70.4% external

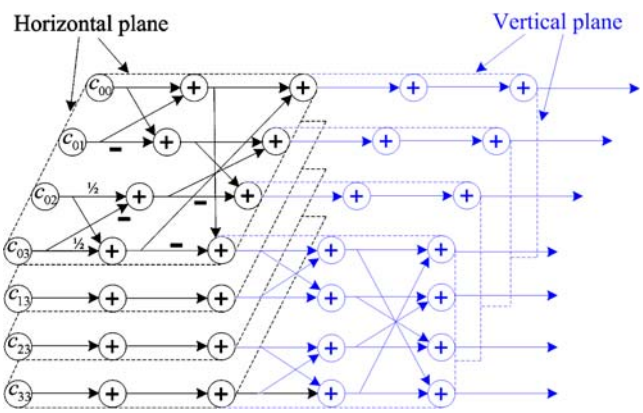


Figure 8 Fast butterfly for direct 2D.

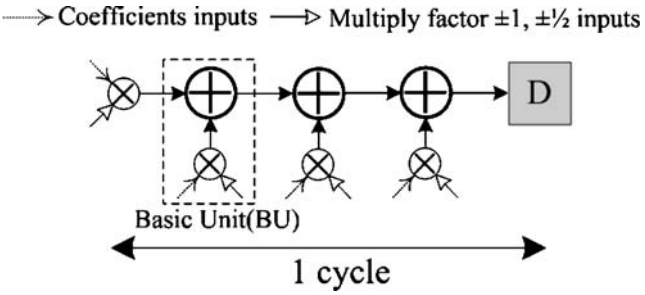


Figure 9 Basic serial approach.

memory bandwidth reduction for 4×4, 8×8, and 16×16 partitions, respectively.

Extended 2× 2 raster order to reduce memory access, it is necessary to increase data reuse probability for overlapped regions of neighboring interpolation window. An extended 2×2 raster scan order is able to save 30% of access times at the cost of 6×9 pixel buffers [6].

TAG method as aforementioned, a substantial amount of MBs inside a video sequence are predicted at relatively larger partitions, i.e. 16×16, 8×16 or 16×8. A TAG is thus added during the MV access of each partition, which indicates current partition size and avoids redundant MV access for large partitions [17]. The memory access is reduced by 71.9% with negligible hardware overhead.

Interleaved chroma access The memory organization of luma and chroma data is also crucial. The luma and chroma are usually stored separately since they are processed independently. For chroma storage, Cb and Cr components can be arranged in a separated [11] or interleaved fashion [7]. Simulation results show that the interleaved method costs 11% less hamming distance, which translates into 11% less memory address switches without any area/throughput penalty. Furthermore, interleaved chroma also increases external SDRAM access operations from 2.18~2.75 words/burst to 4.36~4.38 words/burst when reference Cb & Cr blocks are fetched with the same MV, which greatly reduces external memory latency.

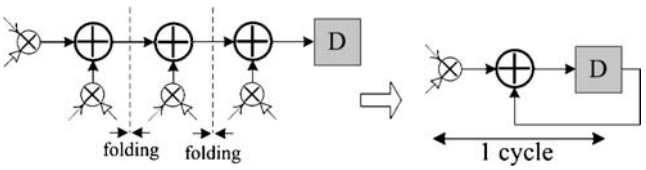


Figure 10 Folding approach.

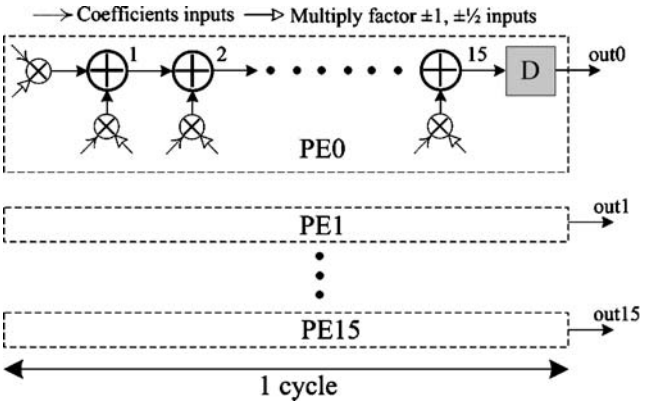


Figure 11 Basic 16-parallel direct 2D implementation.

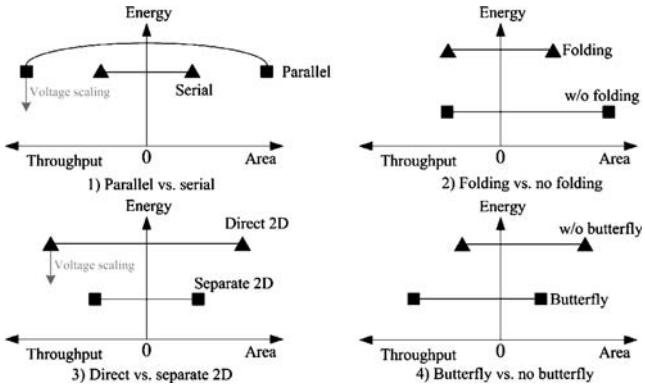


Figure 12 Energy-throughput-area characterization of different approaches.

3.2.3 Interpolator

The luma interpolation in H.264/AVC is a complex 2D filter process which can be decomposed into vertical filtering and horizontal filtering individually. Predicting pixels in serial degrades system performance. Pipelining and parallelism are indispensable to meet real-time decoding requirement. A proposed implementation requires nine horizontal 6-tap filters and four vertical 6-tap filters, as well as four bilinear filters for luma prediction. Vertical and horizontal filtering can take place simultaneously if there is no data-dependency. However, bilinear filtering always lags one cycle behind since it needs output results from vertical and/or horizontal filters. A pipelined parallelized prediction flow of pixel "j" is illustrated in Fig. 15.

Although the chroma interpolation algorithm is different from luma, the interpolator can be shared since chroma interpolation is carried out after luma interpolation. The combined architecture reduces the gate count by 20% [18] as compared with separate solution.

Table 12 summarizes the approaches for PTA optimization of inter prediction. VBS/VBSH increases area a little bit since it needs additional logics to identify the block shape. Although a large buffer increases data reuse and thus improves throughput and reduces external memory access power, the buffer area is increased accordingly. The same applies to extended 2×2 raster scan order. The TAG method also increases area due to the additional 1bit TAG for each partition. Pipelining and parallelism of interpolator help to reduce the power only when the supply voltage can be scaled down.

As can be seen, most of the proposed techniques reduce the power consumption and increases throughput at the cost of large area. Since the inter prediction is identified with

Table 11 IT implementation comparison.

				Relative energy units	Latency		Area	
					# of cycles	Critical path delay	# of adders	# of reg.
Serial	Separate 2D		no folding	320	32	3	3	16
			3-folding	448	96	1	1	17
	Direct 2D		no folding	720	16	15	15	0
			15-folding	1168	240	1	1	1
Parallel	4-parallel separate 2D	w/o butterfly	no folding	320	8	3	12	16
			3-folding	448	24	1	4	20
		with butterfly	no folding	224	8	2	8	16
			2-folding	288	16	1	4	20
	16-parallel direct 2D	w/o butterfly	no folding	720	1	15	240	0
			15-folding	1168	15	1	16	16
		with butterfly	no folding	192	1	4	64	0
			2-folding	224	2	2	32	16
			4-folding	288	4	1	16	16

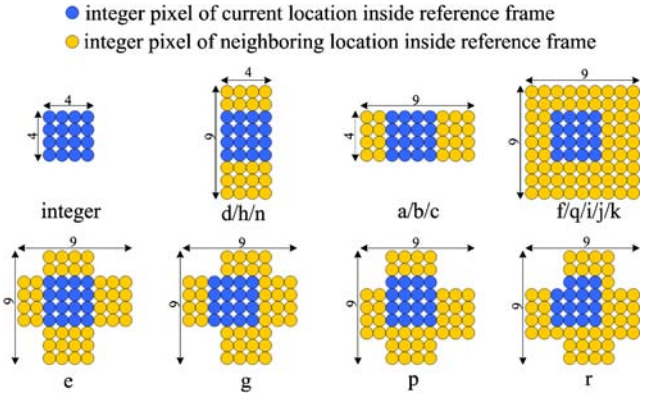


Figure 13 VBSH for 4x4 block prediction (pixel position a ~ r are specified in [1]).

large power consumption and system bottleneck [33], it is preferable to trade chip area for power and throughput.

3.3 Deblocking Filter

As compared to the filters used in previous video standards such as MPEG-2 or H.263, the in-loop deblocking filter of H.264/AVC is much more complex due to higher adaptability of filter process and smaller basic processing block size. It is usually another bottleneck of system's throughput. The design of deblocking filter should focus on the optimization of critical issues like minimizing cycle/MB and memory access.

3.3.1 Pipelining

Pipelining the deblocking filter can greatly improve the throughput. The average time required for one filter operation is reduced significantly when the tasks are distributed among several nearly balanced stages. A five-stage pipeline example is illustrated in Fig. 16.

3.3.2 Reconfigurable Datapath

In the H.264/AVC standard, operations for different Boundary Strength (BS) are implemented by 3-, 4- or 5-tap

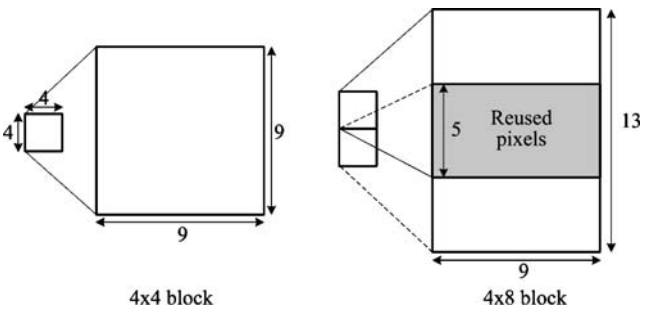


Figure 14 Increased buffer size.

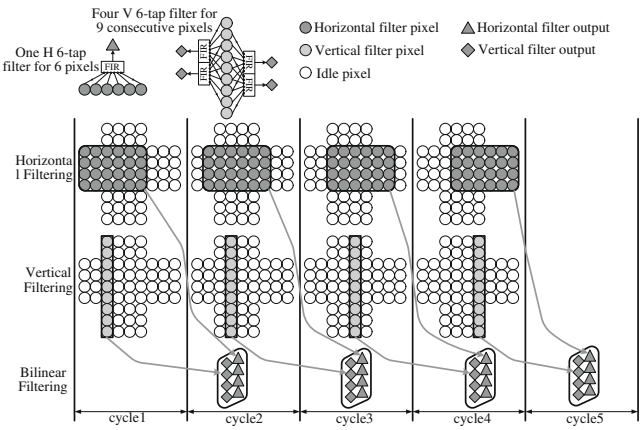


Figure 15 Pipelining and parallel prediction of pixel "j".

filters. Although distinct filters are utilized to filter pixels in different positions, we can still make hardware reuse by sharing common terms and by re-configuring datapath to suit various operations. A parallel-in parallel-out (eight pixels in, eight pixels out) reconfigurable datapath is described in Fig. 17.

Compared with traditional designs, the number of adders has been reduced by 2.46 times, from 32 to 13 [32]. Other function units such as shift and round have also been reused while the number of clip operation remains. The area overhead is 17 mux which can be easily compensated by the reduction of the adders.

3.3.3 Hybrid Order and Memory Organization

The memory organization and the edge filter order are crucial factors for deblocking filter's throughput. According to H.264/AVC standard, nearly all decoded pixels need to be loaded to the deblocking filter which poses a heavy burden on memory access. Furthermore, pipelined architecture introduces overlapped memory operations at different pipeline stages which require double memory bandwidth. To reduce the memory access as well as the memory cost, memory organization and edge filter order should be co-designed to achieve maximum performance.

A hybrid edge filter order (Fig. 18) and its associated memory organization (Table 13) are able to achieve 204 cycles/MB throughput with only single-port SRAM, which satisfies both high throughput and low area cost requirements.

3.3.4 BS-Aware

Whether the current edge needs to be filtered or not depends on the Boundary Strength (BS), which is highly correlated to video characteristics and coding parameters.

Table 12 Inter prediction techniques summary.

		Power	Throughput	Area
Adaptive pipeline		↓	↑	–
VBS/VBSH		↓	↑	↑
Buffer reuse		↓	↑	↑
Extended 2×2 raster scan		↓	↑	↑
TAG for MV prediction		↓	–	↑
Interleaved chroma access		↓	↑	–
Interpolator	Pipelining & parallelism	↓	↑	↑
	Combined luma/chroma	–	–	↓

BS equals 0 means no filtering and the filtering process can be totally skipped. Therefore, the deblocking filter should be BS-aware to reduce the redundant data transfers between reconstructed memory and the filter core. Simulation result [26] shows that generally there are 60% of edges with BS = 0. With BS-aware, all the data transfer and filtering for these edges can be eliminated.

Table 14 summarizes the PTA optimization of deblocking filter. Pipelining helps to reduce the power consumption only when the supply voltage scales down. By carefully manage the filter order and memory organization, all three aspects of PTA can be optimized.

4 Implementation Guidelines

Both the system level and block level design techniques are analyzed in previous two sections. There are several design tradeoffs in terms of power, throughput, and area. There is no fixed solution that achieves best PTA result for all cases. Designers usually need to trade among various performance parameters to gain near-optimal design for a specific application.

Based on the above analysis and our design experiences of five dedicated decoder chips, we derive guidelines for the ASIC design of H.264/AVC decoder.

1) For low-resolution decoders[5], 4×4 pipeline is a good compromise between throughput and area, while for high-resolution designs like HDTV level [6, 8], a hybrid pipeline architecture is more preferable since it reduces the chip area while satisfying the throughput requirement. A pure 16×16 coarse granularity is not recommended because some function units, like IT and intra prediction, will be over-designed in terms of area and throughput.
2) The parallelism of 4 pixels is preferable for most of the building blocks, such as inverse transform and prediction. 1×4 column method [5, 6] suits joint PTA optimization better as compared with 4×1 row method.
3) Memory organization is crucial in joint PTA optimization of entire system. The FoC scheme [5] is able to achieve the smallest power consumption if the I/O and SDRAM power is larger (5~10 times) than the corresponding on-chip SRAM power. The NoC scheme [7] achieves the smallest area but needs to spend a substantial amount of energy for memory access. The PoC scheme [6] is a compromise between them which needs careful analyses through simulation to determine the size of the on-chip memory.
4) There are several solution pairs of mutually exclusive techniques for IT block, normal vs. fast butterfly, separate 2D vs. direct 2D, serial vs. parallel, folding vs. non-folding, etc. Except for butterfly which always outperforms non-butterfly architecture in all cases,

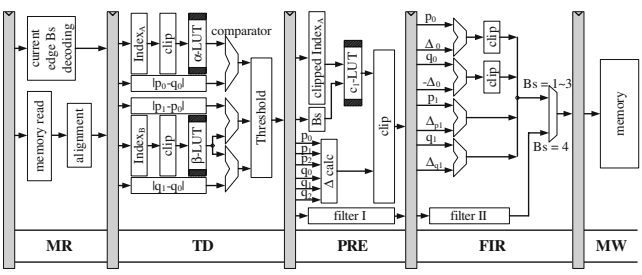


Figure 16 Pipelined deblocking filter.

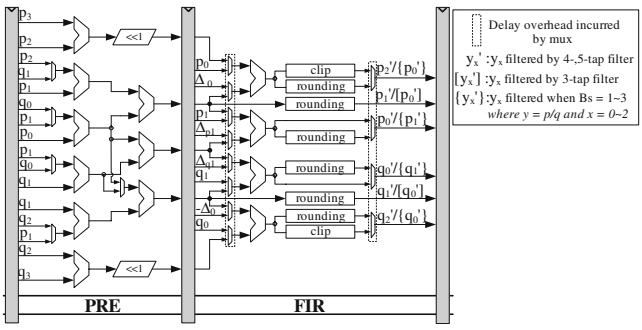


Figure 17 Pipelined reconfigurable datapath.

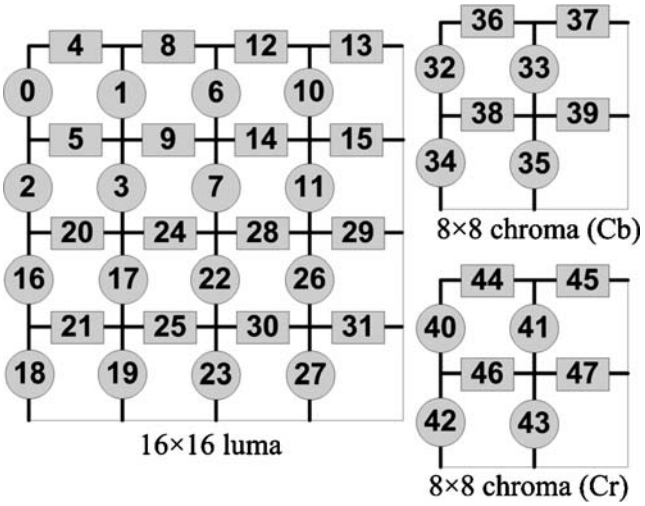


Figure 18 Hybrid edge filter order.

other techniques optimize one or two aspects of PTA. The 16-parallel direct 2D method is able to satisfy realtime decoding of high resolution video sequences with certain area penalty, while the 4-parallel separate 2D approach is more preferable for low-resolution or mobile applications since it is a good balance regarding PTA.

5) The memory bandwidth is the bottleneck of inter prediction. Both VBS [7] and VBSH [5] are able to substantially reduce external memory bandwidth with small area penalty. The size of on-chip buffer is also crucial. The larger the buffer, the less number of redundant memory access can be achieved. Although large on-chip buffer consumes additional chip area, as the technology shrinks, most of designs can afford this amount of added hardware for less power and higher throughput. Therefore, we suggest using 13×13 [5] pixel buffers for area-critical applications, while 13×21/21×13 [8] or even 21×21 for high performance applications.
6) Unlike inter prediction where reference pixel fetch and interpolation can be regarded as two separate processes, the architectural aspects of a deblocking filter, like pipeline architecture, filtering order, and memory organization should be co-designed since all of them may affect the performance significantly. Generally, hybrid edge filter order [5–9] helps to increase neighboring data reuse; proper memory organization helps to reduce memory bandwidth; pipelining [5] favors higher throughput but increases area and introduces pipelining hazards that should be carefully managed.

Table 14 Deblocking filter techniques summary.

	Power	Throughput	Area
Pipelining	↓	↑	↑
Reconfigurable datapath	↓	–	↓
Hybrid edge filter order	–	↑	–
Memory organization	↓	↑	↓
BS-aware	↓	↑	–

5 Conclusion

This paper mainly deals with the power, throughput, and area optimization of H.264/AVC decoder. The optimization starts at system level and then traverses to block level. At system level, we showed how three most important architecture factors, pipelining, parallelism, and memory organization, can affect the decoder in terms of PTA. The 16×16 granularity pipeline is characterized with highest performance, while 4×4 pipeline benefits from smallest intermediate pipeline registers. 4-pixel level parallelism is a compromise between hardware cost and system performance, and it's better to adopt hybrid 4×1 row and 1×4 column method for the parallelism of different building blocks. We also compared different memory schemes, and proved that if IO power and SDRAM power are considerable larger (>5time, which is usually the case for modern

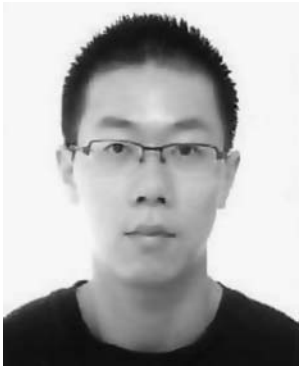
Table 13 Memory organization.

Purpose	Size (bit)			Type
	Luma	Chroma	Total	
Left neighboring Memory (L_RF)	512	512	1024	Two-port register file
Upper neighboring memory(U_RAM)	32 W	32 W	64 W	Single-port SRAM
Current MB memory(C_RAM)	2048	1024	3072	Single-port SRAM
Transposition buffer(T0 ~ T6)	7×128 = 896			DFFs

technology) than SRAM power, FoC scheme is the best low-power solution. At block level, optimization techniques for several main building blocks are discussed in great detail. There is no single solution to achieve an optimal PTA for all cases. Designers usually need to evaluate and to balance among conflicting requirements and choose a solution most suitable for the current application. A set of guidelines are derived to direct the implementation of decoders in general.

References

1. Team, J. V. (2003). Advanced video coding for generic audiovisual services. *ITU-T Recommendation H.264 and ISO/IEC 14496-10 AVC*, May 2003.
2. Wiegand, T., Sullivan, G. J., Bjontegaard, G., & Luthra, A. (2003). Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology, 13* (7), 560–576.
3. Iverson, V., McVeigh, J., & Reese, B. (2004). Real-time H.264/AVC codec on Intel architectures. *IEEE International Conference on Image Processing, 2*, 757–760.
4. Moshe, Y., Peleg, N. (2005). Implementation of H.264/AVC baseline decoder on different digital signal processors. *47the International Symposium on ELMAR*, pp 37–40, June 2005.
5. Xu, K., Choy, C. S. (2007). Low-power H.264/AVC baseline decoder for portable applications. *International Symposium on Low Power Electronics and Design* 256–261.
6. Liu, T. M., et al. (2007). A 125 μW, fully scalable MPEG-2 and H.264/AVC video decoder for mobile applications. *IEEE Journal of Solid-State Circuits, 42*(1), 161–169.
7. Lin, C. C., et al. (2007). A 160 K gates/4.5 KB SRAM H.264 video decoder for HDTV applications. *IEEE Journal of Solid-State Circuits, 42*(1), 170–182.
8. Chien, C. D. et al. (2007). A 252kgate/71 mW multi-standard multi-channel video decoder for high definition video applications. *IEEE ISSCC Dig. Tech*. Papers, 282–283.
9. Liu, T. M et al. (2005). An 865-μW H.264/AVC video decoder for mobile applications. *in Proc. IEEE Asia Solid-State Circuits Conference*, 301–304.
10. Park, S., Cho, H. J., Jung, H., Lee, D. D. (2005). An implemented of H.264 video decoder using hardware and software. *IEEE Custom Integrated Circuits Conference*, 271–275.
11. Kang, H. Y., Jeong, K. A., Bae, J. Y., Lee, Y. S., Lee, S. H. (2004). MPEG4 AVC/H.264 decoder with scalable bus architecture and dual memory controller. *IEEE International Symposium on Circuits and Systems*, 145–148.
12. Hu, Y., Simpson, A., McAdoo, K., Cush, J (2004). A high definition H.264/AVC hardware video decoder core for multimedia SoC's. *IEEE International Symposium on Consumer Electronics*, 385–389.
13. Chen, T. C., Lian, C. J., Chen, L.G. (2006). Hardware architecture design of an H.264/AVC video codec. *Asia and South Pacific Conference on Design Automation*, 750–757.
14. Fujiyoshi, T., et al. (2006). A 63-mW H.264/MPEG-4 audio/visual codec LSI with module-wise dynamic voltage/frequency scaling. *IEEE Journal of Solid-State Circuits, 41*(1), 54–62.
15. Lian, C. J., Tseng, P. C., Chen, L. G. (2006). Low-power and power-aware video codec design: an overview. *China Communications*, 45–51.
16. Lie, W. N., Yeh, H. C., Lin, T. C. I., & Chen, C. F. (2005). Hardware-efficient computing architecture for motion compensation interpolation in H.264 video decoding. *IEEE International Symposium on Circuits and Systems, 3*, 2136–2139.
17. Xu, K. (2007). *Power-efficient design methodology for video decoding*. PhD Dissertation, The Chinese University of Hong Kong, Aug. 2007.
18. Liu, T. M. (2007). *Study of MPEG-2 and H.264 video decoders for mobile applications*. PhD Dissertation, National Chiao Tung University, May 2007.
19. Chen, T. W., et al. (2005). Architecture design of H.264/AVC decoder with hybrid task pipelining for high definition videos. *IEEE International Symposium on Circuits and Systems, 3*, 2931–2934.
20. Huang, Y. W., Chen, T. W., Hsieh, B. Y., Wang, T. C., Chang, T. H., & Chen, L. G. (2003). Architecture design for deblocking filter in H.264/JVT/AVC. *International Conference on Multimedia and Expo (ICME'03), 1*, 693–696.
21. Sima, M., Zhou, Y. H., & Zhang, W. (2004). An efficient architecture for adaptive deblocking filter of H.264/AVC video coding. *IEEE Transactions on Consumer Electronics, 50*(1), 292–296.
22. Chang, S. C., Peng, W. H., Wang, S. H., & Chiang, T. H. (2005). A platform based bus-interleaved architecture for de-blocking filter in H.264/MPEG-4 AVC. *IEEE Transaction on Consumer Electronics, 51*(1), 249–255.
23. Sheng, B., Gao, W., & Wu, D. (2004). An implemented architecture of deblocking filter for H.264/AVC. *IEEE International Conference on Image Processing (ICIP'04), 1*, 665–668.
24. Liu, T. M., Lee, W. P., Lin, T. A., Lee, C. Y. (2005). A memory-efficient deblocking filter for H.264/AVC video coding. *IEEE International Symposium on Circuits and Systems*, 2140–2143.
25. Liu, T. M., Lee, C. Y. (2007). Design of an H.264/AVC decoder with memory hierarchy and line-pixel-lookahead, *Journal of VLSI Signal Processing*.
26. Lou, J., Jagmohan, A., He, D., Lu, L. G., Sun, M. T. (2007). Statistical analysis based H.264 high profile deblocking speedup. *IEEE International Symposium on Circuits and Systems*, 3143–3146.
27. Katevenis, M. "On-chip SRAM and power consumption", lecture notes, Department of Computer Science, Unversity of Crete, Greece, http://archvlsi.ics.forth.gr/~kateveni/534/05a/s21_chip.html
28. Malvar, H. S., Hallapuro, A., Karczwicz, M., & Kerofsky, L. (July 2003). Low-complexity transform and quantization in H.264/AVC. *IEEE Transactions of Circuits and Systems on Video Technology, 13*, 598–603.
29. Gordon, B. M. (1996). *Low power video compression for protable applications using subband coding*. Ph.D. dissertation, Stanford University, 1996.
30. Xu, K., Choy, C. S. A power-efficient and self-adaptive prediction engine for H.264/AVC decoding. *IEEE Transactions on VLSI Systems*, accepted for publication.
31. Joint Video Team (JVT) reference software JM, available: http://iphome.hhi.de/suehring/tml/download/.
32. Xu, K., Choy, C. S. A 5-stage Pipeline, 204 Cycles/MB, Single-port SRAM Based Deblocking Filter for H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, accepted for publication.
33. Lappalainen, V., Hallapuro, A., & Hamalainen, T. D. (2003). Complexity of optimized H.26 L video decoder implementation. *IEEE Transactions of Circuits and Systems on Video Technology, 13*, 717–725.



Ke Xu was born in Chengdu, China. He received the B.S. and M.S. degrees in Electrical & Electronic Engineering Department from Fudan University, Shanghai, China, in 2000 and 2003, respectively, and the Ph.D. degree from the Department of Electronic Engineering, The Chinese University of Hong Kong, Hong Kong, in 2007.

From March to August 2001, he was an intern with Fudan Microelectronics Co., Ltd. From 2003 to 2004, he served as a Design Engineer in Alchip Technology. From 2008 to 2009, he was the Staff R&D Engineer in IBM. Currently he is a post-doctoral research fellow in Electrical and Computer Engineering Department, University of Toronto, Toronto, Canada. His research interests include computer architecture, multimedia codec and associated low-power architectures, FPGA design and VLSI implementation.

Tsu-Ming Liu was born in I-Lan, Taiwan, R.O.C. in 1980. He received the B.S. degree in Electronics Engineering from National Chiao-Tung University, Taiwan, in 2002, and the M.S. and Ph.D. degrees from National Chiao-Tung University, Taiwan, in 2004 and 2007, respectively. From July to October 2004, he was an intern in Sunplus Inc.. From 2004 to 2006, he served as a Lecturer in the Tze-Chiang Foundation of Science and Technology (TCFST). In 2007, he joined the MediaTek Inc., where he is currently a senior engineer. In the past years, he has authored and coauthored over 30 papers in academic journals and national/international conference proceedings. His major research interests include binary shape coding, joint source and channel design, H.264/AVC video decoding, and associated VLSI architectures. Dr. Liu received the Best Impact Award from IEEE Taipei Section. He was a recipient of the internal Ph.D. candidate scholarship of MediaTek Inc. in 2007, and he is an honorary member of Phi-Tau-Phi.

Jiun-In Guo was born in Kaohsiung, Taiwan, R.O.C. in 1966. He received the B.S. degree and Ph.D. degree in electronics engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1989 and 1993, respectively. He is currently a Research Distinguished Professor and Chair of the Department of Computer Science and Information Engineering, National Chung-Cheng University, Chiayi, Taiwan.

Prof. Guo was the recipient of the National Science Council (NSC) Research Award in 1996 and 2001. He was the recipient of the 2003 MXIC Young Professor Award for his contributions to the course of low-power Multimedia/DSP Silicon IP Design. He was also the recipient of the 2004 Chinese Institute of Electrical Engineering (CIEE) Outstanding Youth Electrical Engineer Award and the recipient of the 2008 CIEE Tai-Chung Section Outstanding Engineering Professor Award to recognize his excellent contributions to R&D and service of electrical engineering. He was also the recipient of the 2006 Outstanding Research Award of National Chung Cheng University. He has published over 140 technical papers on the research areas of low-power and low cost algorithm and architecture design for DSP/Multimedia signal processing applications. His research interests include image, multimedia, and digital signal processing, VLSI algorithm/architecture design, digital SIP design, and SOC design.

Chiu-sing Choy received his B.Sc., M.Sc. and Ph.D. from the University of Manchester in 1983, 1984 and 1987 respectively, major in electrical and electronics engineering. From 1985, he spent a year in Ferranti Microelectronics, Oldham, U.K., participating in ASIC technology research. In 1986, he joined Department of Electronic Engineering, The Chinese University of Hong Kong, where he is presently a Professor.

Dr. Choy is interested in Network-on-Chip, mixed signal designs, RFID technology and applications, Structured ASIC, ultra-low supply circuit techniques, multimedia and baseband processing.